REMARKS

Claims 1-5 and 21-35 are pending and stand rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 6,151,602 to Hejlsberg et al. (hereinafter, Hejlsberg). Applicants traverse the rejections and believe all claims are in condition for allowance. Therefore, Applicants respectfully request reconsideration and withdrawal of all rejections under 35 U.S.C § 102(b).

Without limitation to the claims, embodiments of the present invention relate to a data engine of a Programmable Streaming Data Processor (PSDP) which is arranged to perform primitive functions directly on database records. A data parser in the data engine can be programmed to recognize the record and field structures of non-field delineated data received from a source, such as a mass storage device. The data parser then parses non-field delineated data into field delineated data under instruction from an external processing unit. The data engine employs logical arithmetic methods to compare fields with one another, or with values otherwise supplied by general purpose processors, to precisely determine which records are worth selecting to be transferred to memory for further processing by the more general purpose distributed Job Processing Units (JPUs). An output tuple is then formed comprised of the fields of the source record of the disc that are to be selected for further processing by the CPU and PSDP. Thus, the data engine allows the PSDP to perform certain preliminary processing in order to reduce the computational load on the local CPU.

Hejlsberg relates to a self-describing data packet that may be transmitted between a client and a server in a three-tier data processing system. The self-describing data packet allows platform- and protocol-independent communication between clients and servers based on disparate platforms employing various protocols, as has become common with the advent of communications over the Internet.

An example three-tier data processing system 300 shown in FIG. 3 comprises a client 310, a server 350 and a middle tier, which includes a provider 320 and a resolver 325. With reference to FIG. 3 (in conjunction with the flow diagram 500 of FIG. 5), as described at col. 6, lines 36-58, to obtain data, the client 310 sends a request (501) for data from a data source to the middle tier (provider) 320 which, in turn (502), retrieves the data from the server. The request is

honored by the provider 320 generating and returning a data "snapshot" of the result data set (503-507), which is stored locally 315 at the client in the form of a self-describing data packet, as described in col. 3, lines 7-16 and 27-45, that represents a results set received by a client from a data source.

The self-describing packet 400, as illustrated in FIG. 4 and described at col. 8, lines 14-23 and 36-48, includes a header 410, metadata including column descriptors 421, and rows 430 containing the actual row data retrieved from the database server. The column descriptors 421 fully characterize a particular column, such as by providing the column name and data type. The row data 430 includes the actual data of the data set.

Referring back to FIG 3 (in conjunction with the flow diagram 600 of FIG. 6), as described at col. 7, lines 39-54, the structure of the data in a self-describing data packet is described using metadata. Upon receiving the self-describing data packet from the provider 320, the client 310 unpacks (601) the data. Through use of the metadata (602, 603), the client has full knowledge of the data set. The data then is stored locally (604) and processed as if it originally were local data (605). Now the data exists at the client and the data packet may be discarded.

Once the data is at the client, it is subject to operations, such as insert, delete and modify, which may be stored as delta packets. For example, an insert action creates an insert record indicating those records which have been inserted at the client, a delete action creates a record which includes the original data so the resolver can locate the original data record at the back end, and a modify action causes the system transmit the original data record and a new modified data record. The resolver 325, upon receiving a delta packet containing an operation, applies logic for effecting the user-specified modifications to the data set present at the server 350.

However, the data in Hejlsberg is not non-field delineated as required by the independent claims. Quite to the contrary, the data in Hejlsberg is field-delineated at all times at (1) the server, (2) in its packetized form for transmission, (3) and as unpacked at the client.

First, for example, the provider 320 retrieves field-delineated data from a database server (e.g., SQL database tables). That field-delineated data is then transformed by the provider 320 into a self-describing data packet.

Second, the rows of data in the self-describing data packet are explicitly field-delineated and include the actual row data of the data set, except that the column description information

has been stripped off into metadata for reconstitution at the client. For example, col. 20, lines 48-52 and 64 through col. 21, line 1 of Hejlsberg teaches "the provider can sequentially read column descriptor information from the data source...and then stream out corresponding metadata...Processing of actual data also occurs sequentially, as the information is being written out to the stream. In particular, the system loops through all data records of the result set and writes or streams out the corresponding field values, at step 505. Here, only actual data up is written out."

Third, the row data is still field-delineated at the client where it is unpacked from the self-describing data packet into a reconstituted record to include the column descriptor information. As taught at col. 21, lines 16-26, "After the data packet is transmitted to the client, at step 601, the client starts the unpacking process by reading the data packet header for processing the column descriptors (step 602) and optional parameters (step 603). With this information, the client can set up a local data store for receiving the actual data, as indicated at step 604. Thereafter, the client can proceed to process the individual data records, at step 605, using the metadata information for correctly interpreting the stream. In this manner, the data is correctly reconstituted at the client (e.g., in a local data store.)"

Therefore, Hejlsberg fails to teach non-field delineated data as required by the independent claims.

Further, Hejlsberg fails to teach an output tuple generator configured to assemble filtered field delineated data into an output tuple. The Examiner relies on col. 21, lines 25-35 of Hejlsberg, which teaches that the reconstituted database records now exist at the client as if it were a local table. However, the reconstituted database records are exactly the same as the database records as they exist at the server end of the system. Therefore, they are not an output tuple comprising filtered field delineated data. Although filtering of some form may be performed in Hejlsberg, it is only after the reconstituted database records are at the client that they may be manipulated by the client so that the resolver 325 may create a delta data packet to execute changes at the server.

In order to anticipate a claim, a reference must teach each and every element of the claim. For the reasons presented above, Hejlsberg fails to teach the elements of independent Claim 1. Therefore, the rejection of independent Claim 1 is overcome and reconsideration is respectfully

requested. Similarly, with regard to the rejection of independent Claim 21, Hejlsberg does not teach the claimed method that substantially corresponds to the data engine of Claim 1. For these reasons, the rejections of independent Claims 1 and 21 are overcome and reconsideration is respectfully requested.

With regard to the rejections of Claims 2-5 and 22-35, these claims are dependent upon Claims 1 and 21, respectively, and therefore contain the limitations of the respective base claims. For these reasons, the rejections of Claim 2-5 and 22-35 are overcome and reconsideration is respectfully requested.

Information Disclosure Statement

A Supplemental Information Disclosure Statement (SIDS) is being filed concurrently herewith. Entry of the SIDS is respectfully requested.

CONCLUSION

In view of the above amendments and remarks, it is believed that all claims are in condition for allowance, and it is respectfully requested that the application be passed to issue. If the Examiner feels that a telephone conference would expedite prosecution of this case, the Examiner is invited to call the undersigned.

Respectfully submitted,

HAMILTON, BROOK, SMITH & REYNOLDS, P.C.

By _____
Gerald P. Kazanjian
Registration No. 61,699
Telephone: (978) 341-0036
Facsimile: (978) 341-0136

Concord, MA 01742-9133
Date: 3/30/2009